or application that deals with many forms of content such as 2d or 3d content creation tools, or even an HTML or a Blog editor.

[0265] When there is a desire to decouple the systems so that system knowledge is kept to a minimum (or when point to point APIs end up creating a distributed monolith and leaking abstractions).

[0266] The foregoing description of the preferred embodiment of the invention has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. It is intended that the scope of the invention be limited not by this detailed description, but rather by the claims appended hereto.

REFERENCES

[0267] [1] *Pro Git,* by Scott Chacon and Ben Straub, 2nd Edition 2014.

[0268] [2] *TARDiS: A Branch-and-Merge Approach To Weak Consistency,* Natacha Crooks, Youer Pu, Nancy Estrada, Trinabh Gupta, Lorenzo Alvisi, and Allen Clement, Proceedings of the 2016 International Conference on Management of Data

[0269] [3] *Versionable, Branchable, and Mergeable Application State,* David H. Lorenz, and Boaz Rosenan, Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software

[0270] [4] *AWS AppSync Developer Guide,* Amazon Web Services, Inc (2020) (available from docs.aws.amazon.com/appsync/latest/devguide).

[0271] [5] *share.js* (available from github.com/josephg/ShareJS) (2014).

[0272] [6] *Google Cloud Firestore* (Aug. 8, 2020) (available from firebase.google.com/docs/firestore).

[0273] [7] *Decibel: the relational dataset branching system,* Michael Maddox, David Goehring, Aaron J. Elmore, Samuel Madden, Aditya Parameswaran, and Amol Deshpande, Proceedings of the VLDB Endowment Volume 9 Issue 9, May 2016.

[0274] [8] *Supporting branched versions on XML documents,* Z. Vagena, M. M. Moro, V. J. Tsotras, 14th International Workshop Research Issues on Data Engineering: Web Services for e-Commerce and e-Government Applications, 2004.

[0275] [9] *The block chain and the CAP Theorem* by Yaron Goland (Mar. 8, 2017).

[0276] [10] *How Bitcoin Loses to the CAP Theorem* by Paul Kernfeld (15 Jan. 2016).

[0277] [11] *Design and Analysis of Index Structures in Multi Version Data.,* Jouini K., Jomier G., New Trends in Data Warehousing and Data Analysis. Annals of Information Systems, vol 3, 2009.

[0278] [12] *Overlapping B+-Trees: An Implementation of a Transaction Time Access Method,* Theodoros Tzouramanis, Yannis Manolopoulos, and Nikos A. Lorentzos, Data & Knowledge Engineering, 29(3):381-404, 1999.

[0279] [13] *The BT-Tree: A Branched and Temporal Access Method,* Linan Jiang, Betty Salzberg, David B. Lomet, Manuel Barrena Garcia, Proceedings of the 26th International Conference on Very Large Data Bases, 2000.

What is claimed is:

1. A computer-implemented system for managing data, comprising:

(a) a computer having a memory;

(b) a processor executing on the computer;

(c) the memory storing a computer application comprising a set of instructions, wherein the set of instructions, when executed by the processor cause the processor to perform operations;

(d) the computer application providing one or more property sets, wherein:

(i) the one or more property sets comprise one or more property set objects created based on a schema;

(ii) the schema defines a type and a version of the one or more property set objects;

(iii) the one or more property sets are organized hierarchically to form a property tree;

(e) the computer application providing a commit graph, wherein:

(i) the commit graph comprises storage for the one or more property set objects;

(ii) the commit graph is an append only data structure;

(iii) the commit graph comprises a topology of changes between states of the one or more property set objects;

(iv) the commit graph comprises two or more commit nodes organized as parent commit nodes and child commit nodes;

(v) the child commit node comprises a pointer to a parent commit node and changes to one of the property set objects;

(vi) as a new commit node is added to the commit graph, the property tree is formed;

(f) the computer application providing one or more change sets, wherein:

(i) each change set represents a change between two commit nodes of the commit graph;

(ii) each change set tracks changes made on the one or more property set objects within the commit graph;

(iii) the changes specify basic operations that are permitted;

(iv) the basic operations comprise insert, remove, or modify operations that are applied on each state to get to a next state of the one or more property set objects; and

(v) each change set is reversible.

2. The computer-implemented system of claim **1**, wherein:

each commit node comprises metadata; and

the metadata identifies a parent commit node, a creator, and a timestamp.

3. The computer-implemented system of claim **1**, wherein:

the commit graph comprises a merge operation;

the merge operation is represented by the two or more commit nodes as a directed acyclic graph.

4. The computer-implemented system of claim **1**, wherein:

the two or more commit nodes are organized in one or more branches;

each of the one or more branches tracks a history of commits that make up a variation of changes made on the data.